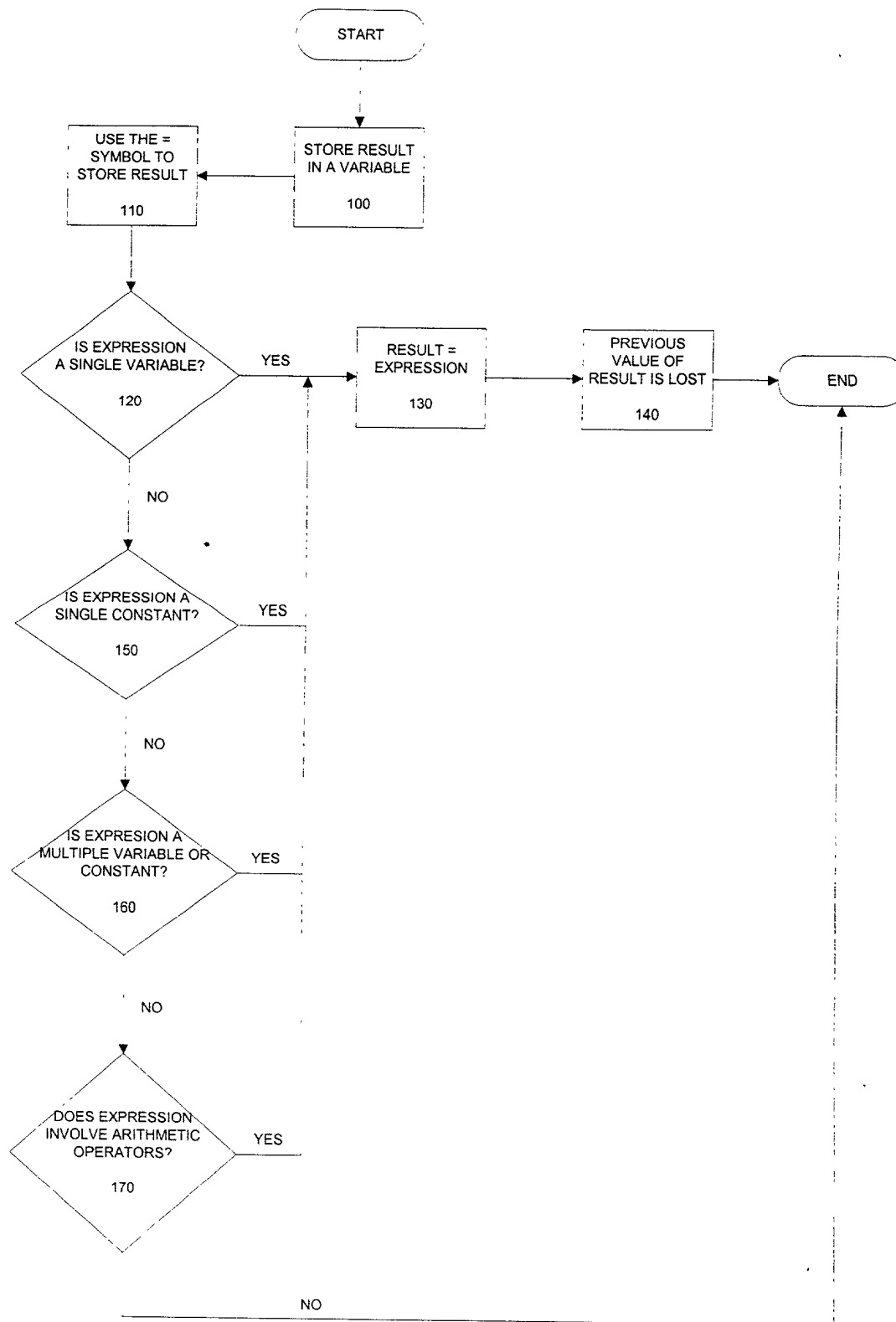
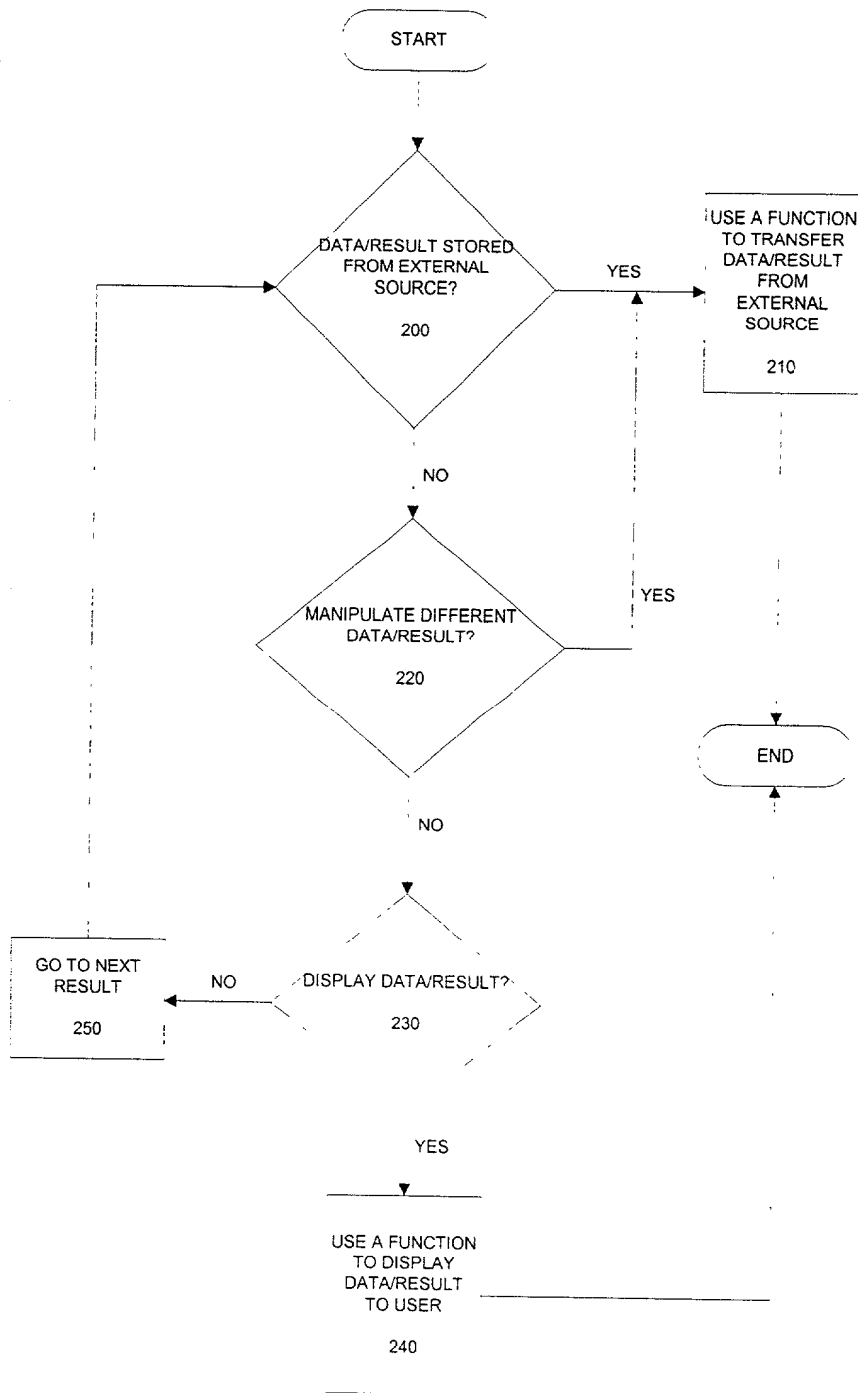


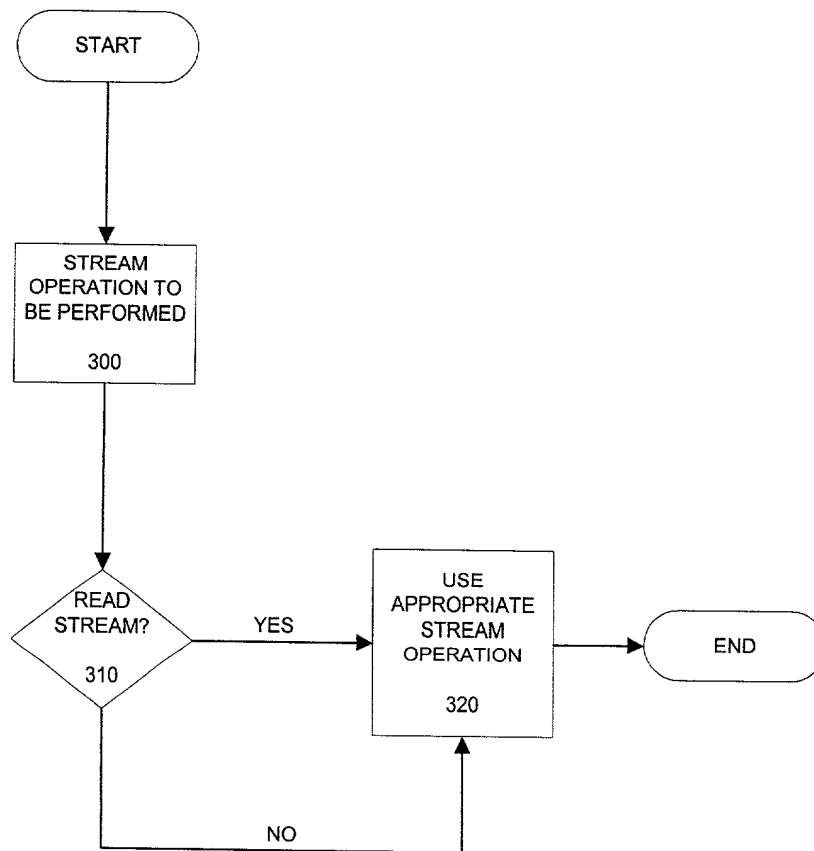
**Figure 1**



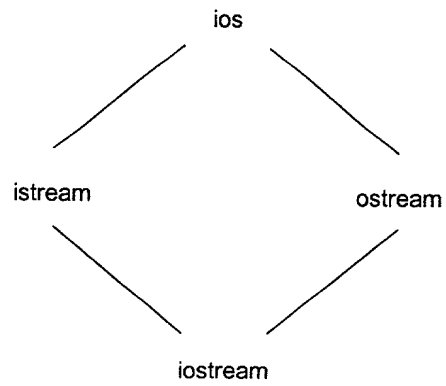
**Figure 2**



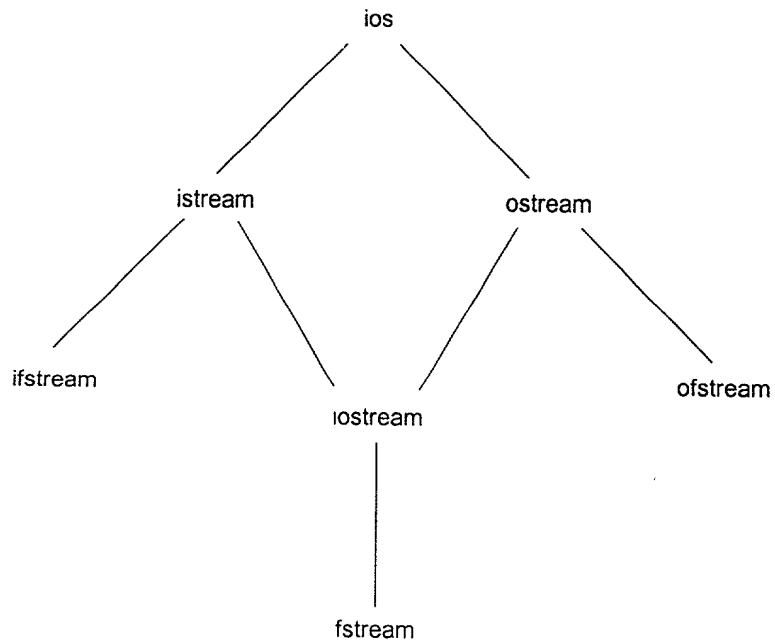
**Figure 3**



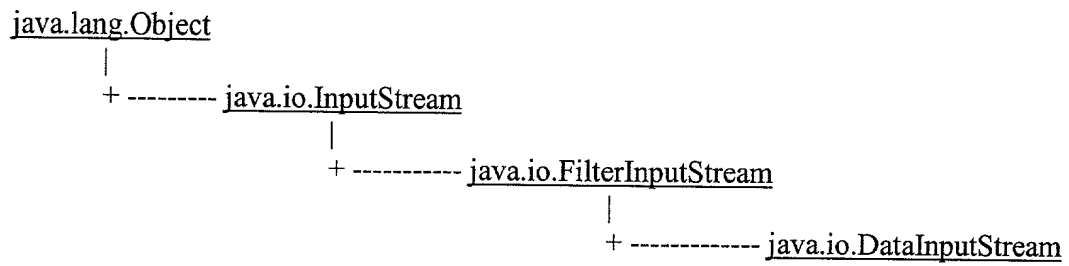
**FIGURE 4A**



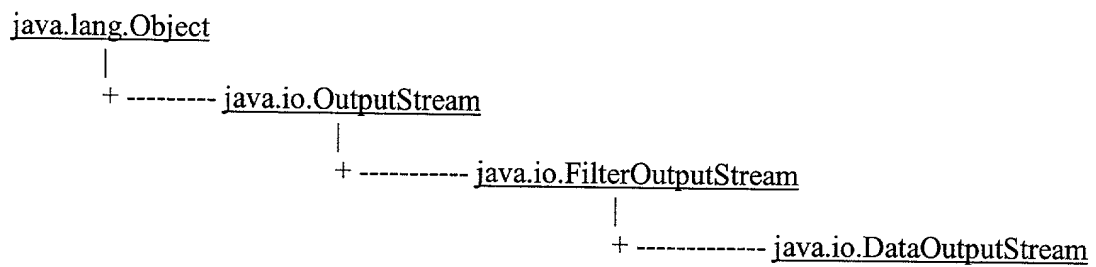
**FIGURE 4B**



**FIGURE 5A**



**FIGURE 5B**



**FIGURE 6A**

Function	Purpose
openin (filename)	Open the named file for input
openout (filename)	Open the named file for output
openup (filename)	Open the names file for update (reading and writing)
openfd (fd)	Open the integer file descriptor as a stream
open (filename, mode)	Open the named file with the given integer mode

**FIGURE 6B**

Function	Meaning
Network.open (addr,port)	Open an active network connection (TCP)
Network.openServer (addr, port, type)	Open a passive network connection (TCP or UDP)
Network.lookupName (name)	Consult a naming service to convert a network name to an address
Network.lookupAddress (ipaddr)	Consult a naming service to convert an integer IP address into a host name
Network.accept()	Wait for and accept an incoming connection
Network.openSocket()	Open a UDP client socket for sending datagrams
Network.send (socket, addr, port, buffer)	Send a UDP datagram to the given socket. The addr and port specify the recipient. The buffer is the string to send
Network.receive (socket, var addr, var port, maxbuffer = 4096)	Wait for an incoming UDP datagram. Blocks until a datagram is received, then returns the data as a string. Also sets the addr and port to the sending address. The maxbuffer argument specifies the max size of data that can be received
Network.peek (socket, var addr, var port, maxbuffer = 4096)	As receive, but don't extract the data from the network. A call to receive() will read the data.
Network.formatIPAddress (addr)	Build a string of the form n.n.n.n out of an integer IP address.

**FIGURE 7**

<b>Data type</b>	<b>Reading</b>	<b>Writing</b>
integer	Decimal integer converted to binary	Decimal integer
string	Whole line read – terminated by line feed character which is discarded	Characters written to stream.
real	ASCII for real value	Written as ASCII
char	Single character	Single character
vector	Each line of file appended to vector - line feed is retained	Each element written
map	cannot read	Elements written as first = second
enumeration constant	cannot read	Name of enumeration constant

**FIGURE 8**

<b>Operation</b>	<b>Result</b>
close (stream)	The stream is closed.
select (stream, timeout)	Returns 1 if there is data waiting to be read from stream. Times out after timeout microseconds.
eof (stream)	Returns 1 if the stream is at the end of file.
flush (stream)	Flush the data remaining in the stream buffers.
getchar (stream)	Read a single character from the stream. Returns the character read.
getbuffer (stream)	Read all the available characters in the stream buffer. Returns a string containing all the characters.
availableChars (stream)	Returns the number of characters in the buffer.
setStreamAttribute (stream, attr, value)	Set the value of a stream attribute.
rewind (stream)	Rewind the stream to the start.
seek (stream, offset, whence)	Move to a new position in a seekable stream.

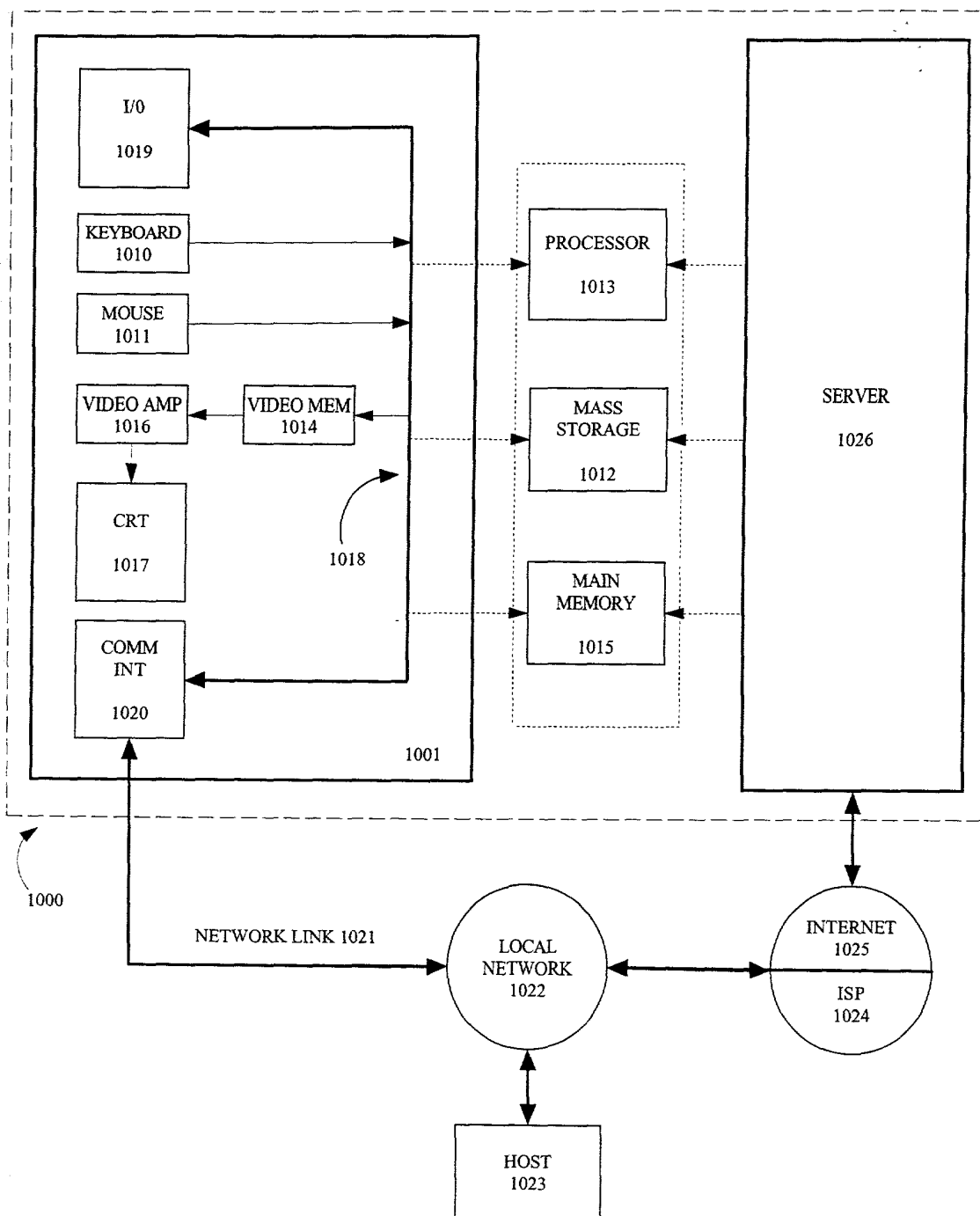
**FIGURE 9 – Page 1**

Output type	Input type	Operation
<b>integer</b>	integer	copy left to right
	real	real converted to integer
	string	string converted to integer if possible, 0 otherwise
	vector	first element converted to integer
	map	first element converted to integer
	char	converted to integer
	block	integer set to address
	enumconst	index into enumeration
	object	call toInteger( ) if present, otherwise address of object
	stream	one integer read from stream
<b>real</b>	integer	converted to real
	real	copied
	string	string converted to real if possible, 0.0 otherwise
	vector	first element converted to real
	map	first element converted to real
	char	converted to integer then real
	block	converted to integer then real
	enumconst	converted to integer then real
	object	toReal( ) called if present, error otherwise
	stream	one floating point number read from stream
<b>string</b>	integer	converted to string
	real	converted to string
	string	copied
	vector	each element appended to string
	map	each element appended to string
	char	converted to string
	block	name of block
	enumconst	name of constant
	object	toString( ) called if present, <u>blockname@address</u> if not
	stream	one line read from stream
<b>char</b>	integer	truncated to 8 bits
	real	runtime error
	string	first character in string
	vector	first element converted to char
	map	first element converted to char
	block	first character of name
	enumconst	'A' = first const, 'B' = second, etc
	object	toChar( ) called if present, error if not
	stream	one char read from stream

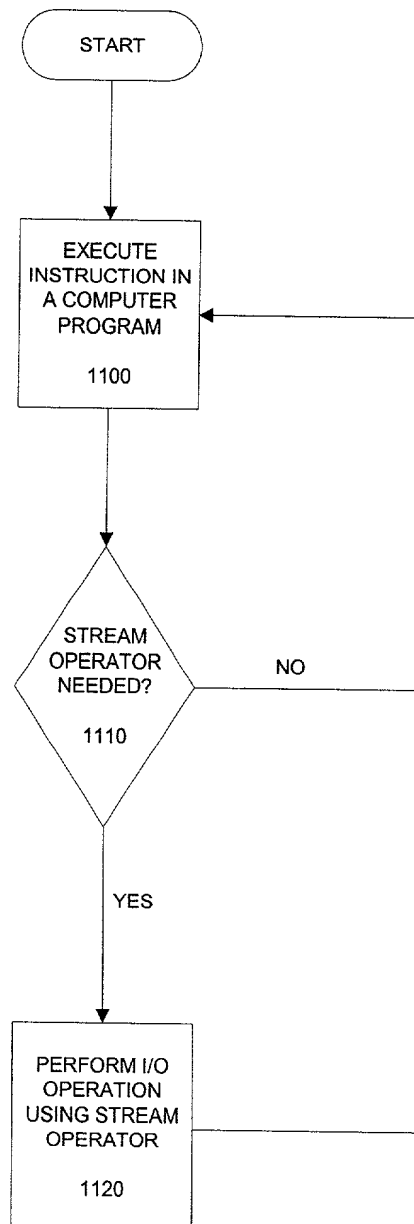
**FIGURE 9 – Page 2**

Output type	Input type	Operation
vector	object	toVector( ) called if present, otherwise object appended to vector
	anything	appended to vector
map	object	toMap( ) called if present, otherwise appended
	anything	appended as {x = x}
function	scalar	function called with single argument
	vector	function called once for each element. Element passed as parameter
	map	function called for each element. Function has to arguments for left and right of map pair.
	stream	function called for each line of input
thread		like function
class		like function only new object created for each
package		like class
enum		runtime error
enumconst		runtime error
object		runtime error
stream	integer	converted to decimal character sequence
	real	converted to floating point character sequence
	string	each character written
	char	single character written
	vector	each element written
	map	each element written as left = right
	block	block name written
	enumconst	name of constant written
	object	“object “ + address written
	stream	steam copied

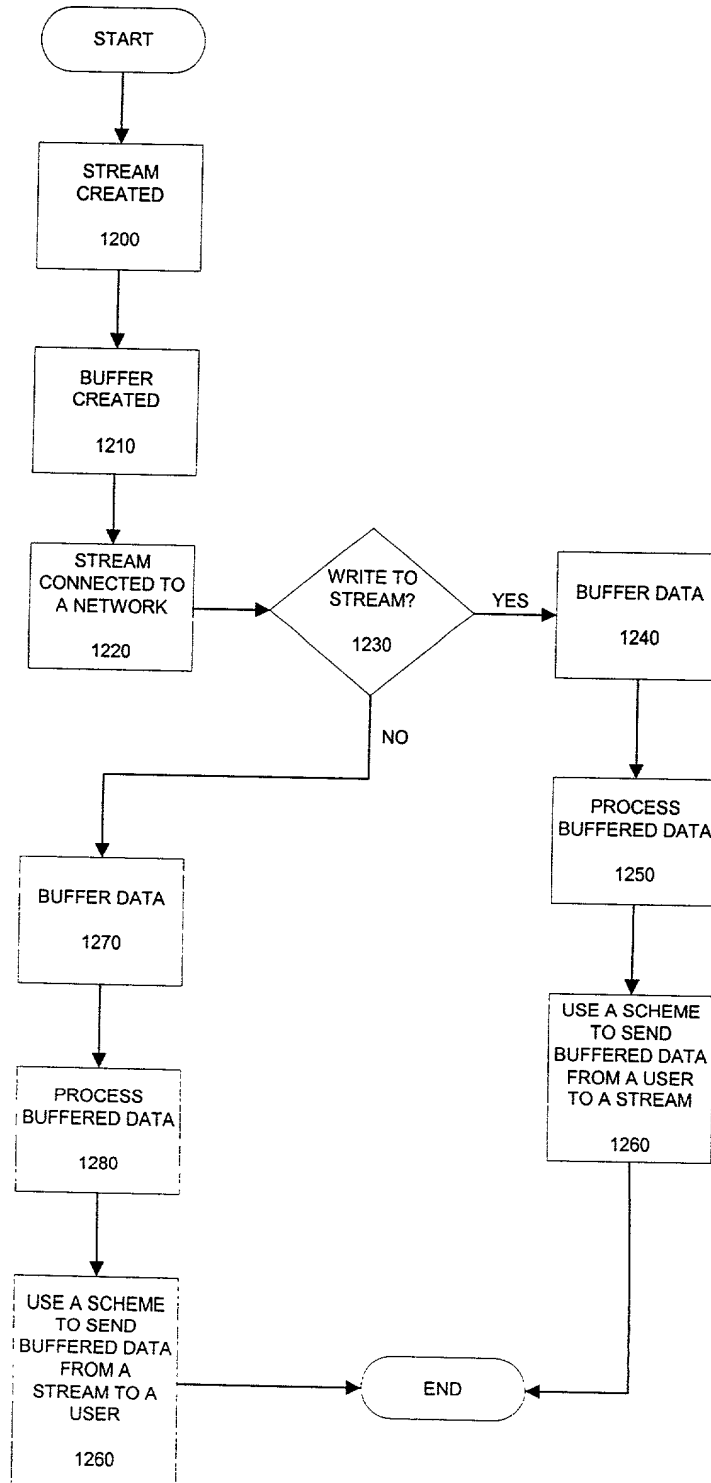
Figure 10



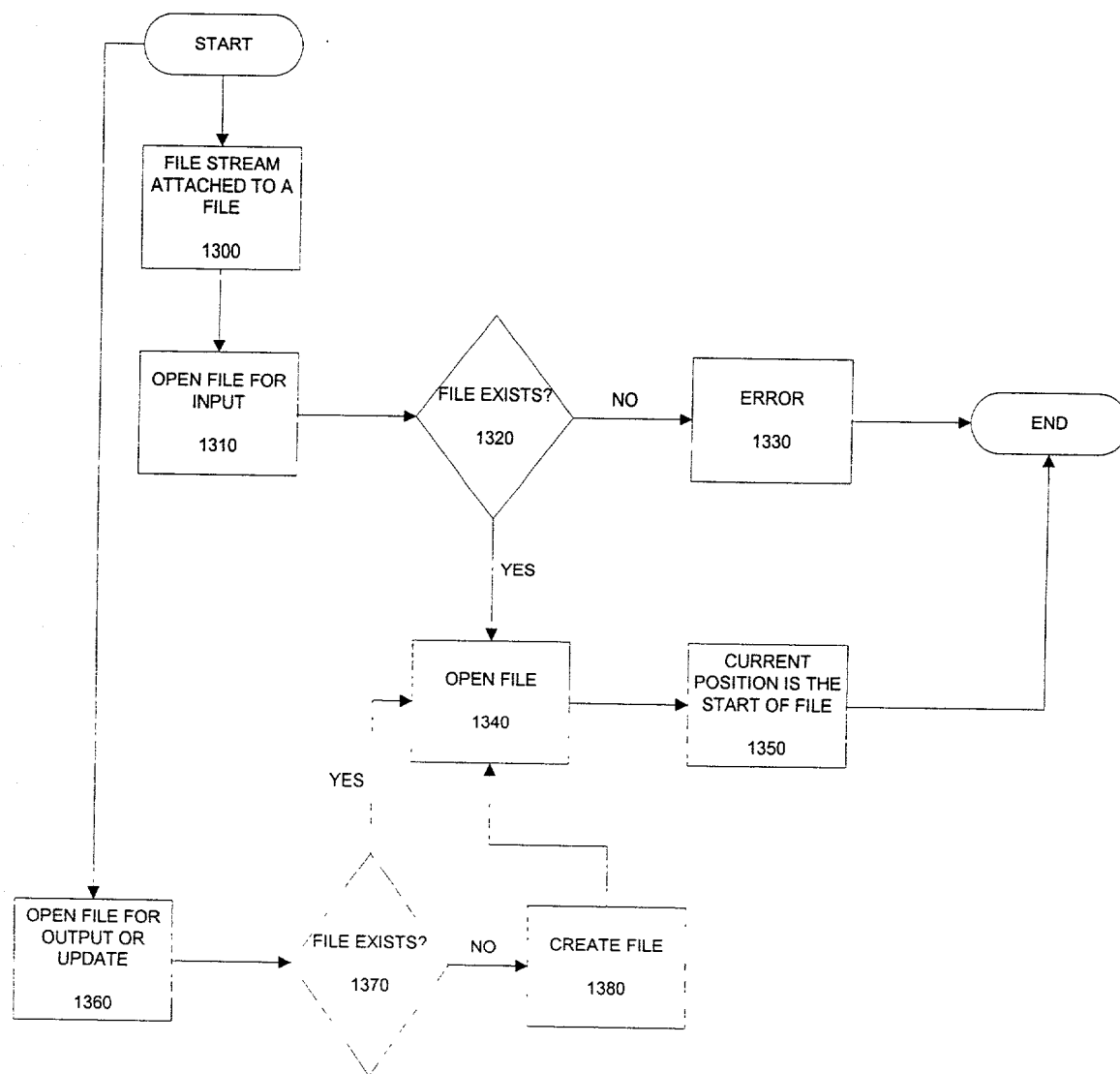
**Figure 11**



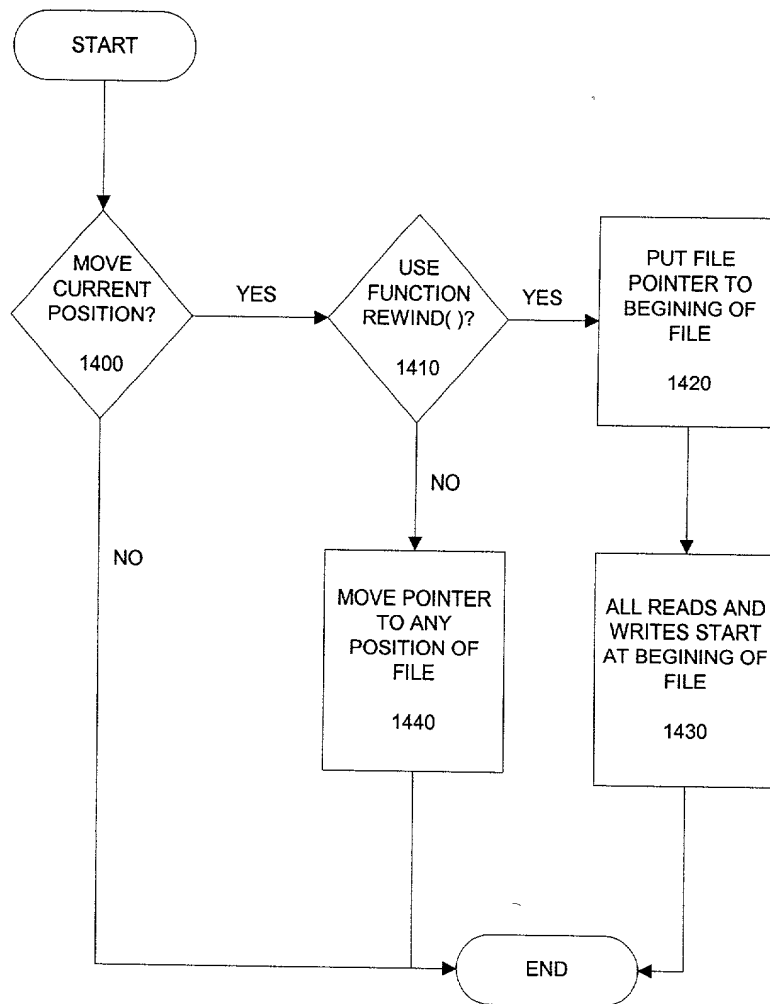
**Figure 12**



**Figure 13**



**Figure 14**



**Figure 15**

